Fondamenti di Java

Soluzione esercizio hashcode

Esercizio

Definire una classe con una hashCode corretta.
Aggiungere delle istanze "uguali" a un set, e controllare la dimensione del set ottenuto. Vi torna il valore ottenuto per la dimensione del set?

Definire una classe con una hashCode non corretta.
Aggiungere delle istanze "uguali" a un set, e controllare la dimensione del set ottenuto. Vi torna il valore ottenuto per la dimensione del set? Potete spiegare quel che osservate?

Implementazione corretta

```
public class Test {
   int a=0;
   Test(int k) {a=k;}
   public boolean equals(Test c){
      if (c==null) return false;
      if (a==c.a) return true;
      return false;
   }
   public int hashCode() {return a;  /* return 0 */ }
   public static void main(String a[]){
      Test k1=new Test(2);
      Test k2=new Test(2);
      Collection col=new HashSet();
      col.add(k1);
      col.add(k2);
      System.out.println(col.size());
   }
}
```

OUTPUT: 1

Implementazione sbagliata

```
public class Test {
   int a=0;
   Test(int k) {a=k;}
   public boolean equals(Test c){
      if (c==null) return false;
      if (a==c.a) return true;
      return false;
   }
   public int hashCode() {return (int)
     (Math.random()*100);}
   public static void main(String a[]){
      Test k1=new Test(2);
      Test k2=new Test(2);
      Collection col=new HashSet();
      col.add(k1);
      col.add(k2);
      System.out.println(col.size());
   }
}
```

OUTPUT: 2

Precisazioni

class Object
equals(Object o)

interface Comparable<T>
compareTo(T o)

interface Comparator<T>
compare(T o1, T o2)
equals(Object o)

Fondamenti di Java

Static

Modificatori: static

Variabili e metodi associati ad una Classe anziche' ad un Oggetto sono definiti "static".

Le variabili statiche servono come singola variabile condivisa tra le varie istanze

I metodi possono essere richiamati senza creare una istanza.

Variabili “static”: esempio 1

```
public class  S {
   static int instanceCount = 0; //variabile “di classe”
   S() {instanceCount++;}

}
public class  A {
   public static void main(String a[]) {
        new A();
   }
   A() {
      for (int  i = 0;  i < 10;  ++i) {
         S instance=new S();
      }
      System.out.println("# of instances:
        "+S.instanceCount);
   }
}
```

Output:

```
# of instances: 10
```

Variabili "static": esempio 2

```
class  S {
   static int instanceCount = 0; //variabile "di classe"
   S() {instanceCount++;}
   public void finalize() {instanceCount--;}
}
public class  A {
   public static void main(String a[]) {
        new A();
   }
   A() {
     for (int  i = 0;  i < 10;  ++i) {
         S instance=new S();
     }
     System.out.println("# of instances:"+S.instanceCount);
     System.gc();
     System.out.println("# of instances: "+S.instanceCount);
   }
}
```

Output:

```
# of instances: 10
# of instances: 0
```

Metodi "static": esempio 1

```
class  S {
   static int instanceCount = 0; //variabile "di classe"
   S() {instanceCount++;}
   static void azzeraContatore() {instanceCount=0;}
}
public class  A {
   public static void main(String a[]) {
        new A();
   }
   A() {
      for (int  i = 0;  i < 10;  ++i) {
         if (i%4==0) S.azzeraContatore();
         S instance=new S();
      }
      System.out.println("instanceCount:
"+S.instanceCount);
  }
}
```

Può agire solo su variabili statiche!

Output:
instanceCount: 2

Ruolo:
Metodi che agiscono su variabili statiche

metodi "static": esempio 2

```
Math.sqrt(double x);
System.gc();
System.arrayCopy(...);
System.exit();
Integer.parseInt(String s);
Float.parseFloat(String s);
```

Notare la maiuscola! (per convenzione)

Ruolo: analogo alle librerie del C

```
Che cos'e':
System.out.println() ?
```

Perchè il main è "static"?

```
public class  A {
    String s="hello";
    public static void main(String a[]) {
        System.out.println(s);
    }
}
```

```
Non static variable s cannot be referenced from static context
```

```
public class  A {
    String s="hello";
    public static void main(String a[]) {
      new A();
    }
    A() {
      System.out.println(s);
    }
}
```

```
hello
```